

# Improving Energy Efficiency of MPTCP for Mobile Devices

Yeon-sup Lim  
School of Computer Science  
University of Massachusetts  
Amherst, MA, USA  
ylim@cs.umass.edu

Yung-Chih Chen  
School of Computer Science  
University of Massachusetts  
Amherst, MA, USA  
yungchih@cs.umass.edu

Erich M. Nahum  
IBM Thomas J. Watson  
Research Center  
Yorktown Heights, NY, USA  
nahum@us.ibm.com

Don Towsley  
School of Computer Science  
University of Massachusetts  
Amherst, MA, USA  
towsley@cs.umass.edu

Richard J. Gibbens  
Computer Laboratory  
University of Cambridge  
Cambridge, UK  
richard.gibbens@cl.cam.ac.uk

## ABSTRACT

Multi-Path TCP (MPTCP) is a new transport protocol that enables systems to exploit available paths through multiple network interfaces. MPTCP is particularly useful for mobile devices, which usually have multiple wireless interfaces. However, these devices have limited power capacity and thus judicious use of these interfaces is required. In this work, we develop a model for MPTCP energy consumption derived from experimental measurements using MPTCP on a mobile device with both cellular and WiFi interfaces. Using our energy model, we identify an operating region where there is scope to improve power efficiency compared to both standard TCP and MPTCP. We design and implement an improved energy-efficient MPTCP, called eMPTCP. We evaluate eMPTCP on a mobile device across several scenarios, including varying bandwidth, background traffic, and user mobility. Our results show that eMPTCP can reduce the power consumption by up to 15% compared with MPTCP, while preserving the availability and robustness benefits of MPTCP. Furthermore, we show that when compared with TCP over WiFi, which is more energy efficient than TCP over LTE, eMPTCP obtains significantly better performance with relatively little additional energy overhead.

## Categories and Subject Descriptors

C.2 [Computer-Communication Networks]: Network Protocols; C.2.1 [Network Architecture and Design]: Wireless communication; C.4 [Performance of System]: Measurement techniques, Modeling techniques

## General Terms

Experimentation; Measurement; Performance

## 1. INTRODUCTION

This paper is an extended version of a paper that has been accepted to ACM AllThingsCellular'14 [10].

Multi-Path TCP (MPTCP) is a new standardized transport protocol that simultaneously enables end hosts to take advantage of multiple network interfaces and utilize path diversity in the network [5, 17]. MPTCP can achieve both higher throughput and greater robustness and availability than standard single-path TCP, all while maintaining compatibility with existing applications. One natural fit for MPTCP is use on mobile devices, such as smartphones, which typically include both cellular and WiFi interfaces.

Applying MPTCP to mobile devices introduces a new concern, namely, the additional energy consumption from operating multiple network interfaces. Mobile devices are frequently constrained by the amount of power available in their batteries. Processing power available on-chip continues to grow exponentially, however battery storage increases slowly by comparison. Thus, power consumption is an important area of research, particularly in mobile devices such as smartphones. In order to utilize MPTCP on mobile devices with limited energy resources, it is important to understand the power consumption behavior of MPTCP to ensure that it is practical.

In this work, we shed light on the energy behavior of MPTCP on smartphones. We seek to address the following questions:

- How much energy does MPTCP consume, compared to single-path TCP over WiFi or LTE?
- Are there environments or scenarios where MPTCP is more energy efficient than single-path TCP? If so, how can we recognize them and take advantage of them?
- Can we improve MPTCP's energy efficiency?

In this work, we examine MPTCP energy consumption behavior via a combination of measurement, mod-

eling, and experimentation. We measure power consumption across a range of scenarios, by varying download/upload size and available path bandwidth. We use these measurements together with a regression approach from [8], and determine conditions under which MPTCP is more energy-efficient than either standard TCP or MPTCP. Informed by this model, we design and implement an energy-aware eMPTCP which we have implemented on an Android platform, the Samsung Galaxy S3.

Our paper makes the following contributions:

- We develop an energy model for MPTCP power consumption derived from experimental measurements taken using MPTCP on a mobile device. Our results show that the model accurately predicts the measured energy consumption of MPTCP with an error less than 17%.
- Using our model, we illustrate the tradeoffs between network performance and energy consumption. In most environments, MPTCP does not improve energy efficiency compared to single-path TCP over WiFi. The high cost of cellular tail energy makes power saving difficult to achieve in MPTCP. However, our model does reveal an operating region where MPTCP is more energy efficient than standard TCP, depending on the available path throughputs and transfer size.
- We design, implement, and evaluate eMPTCP, an improved energy efficient MPTCP, on our Android mobile device. Parameterized through our model, eMPTCP dynamically monitors path characteristics and chooses paths based on energy efficiency. We evaluate eMPTCP under multiple scenarios, considering path quality, dynamic bandwidth changes, radio interference, and mobility. We show that eMPTCP can reduce energy consumption by up to 15% compared with MPTCP while still preserving the availability and robustness benefits of multiple paths, at the cost of slightly longer download times.

Previous approaches in this area have either been simulation-only [13] or have looked at much more restricted operating environments [12, 16]. Our work provides general insight and leverages that understanding to provide a new and improved energy-aware MPTCP.

The remainder of this paper is organized as follows: Section 2 provides the background context for our work. Section 3 presents throughput and energy measurements we use to parameterize a single-path TCP energy model. Section 4 describes and validates our energy model for MPTCP. We introduce our energy-aware MPTCP in Section 5. Section 6 provides our results. After reviewing related work in Section 7, we conclude in Section 8.

## 2. BACKGROUND

### 2.1 Benefits of Multi-path TCP

The benefits of leveraging MPTCP in mobile devices are three-fold:

- *Bandwidth*—By utilizing the available bandwidth of each subflow, a MPTCP connection can achieve higher throughput than a standard TCP connection.
- *Robustness*—Even though connectivity in one network can degrade or disappear through movement, MPTCP offers a seamless TCP connection by using paths (subflows) through another network.
- *Compatibility*—The MPTCP layer is hidden from user applications by providing a standard TCP socket. Existing applications using TCP need not to be modified to support MPTCP.

### 2.2 3G/4G State Machine

The 3rd Generation Partnership Project (3GPP) standard defines a state machine for the 3G/4G communication interface, which describes the possible power states of each device connected to the network. To initiate a packet transmission, a 3G/4G communication interface has to switch from a low power to high power state so that packet can be sent or received. If there is no further packet transmission for a period of time, to save energy, the 3G/4G interface returns to a low power state. In the rest of this paper, the terms *promotion* and *tail* are used to refer to the transition periods from a low power state to a high power state and from a high power state to a low power state, respectively [1, 8].

### 2.3 Operation Modes in MPTCP

MPTCP has three modes of operation to control subflow usage. One of them is the backup mode, where MPTCP opens TCP subflows over all interfaces, but uses only a subset of them for packet transmission. If a user sets a particular interface to backup mode, MPTCP sends no traffic through the corresponding subflows unless all other subflows break. By setting up the mode of each interface, a user can manually decide path usage considering traffic pricing or battery life [12]. However, MPTCP does not have any automatic mechanism to control the mode of each subflow in terms of energy efficiency and performance. In this paper, we develop an energy efficient path usage controller while MPTCP operates in Full-MPTCP mode (the regular MPTCP operation in which all subflows are available for use).

## 3. SINGLE-PATH TCP ENERGY MODEL

In this section, we develop an energy consumption model of single-path TCP for our mobile device. The energy model differs between mobile devices. We leave

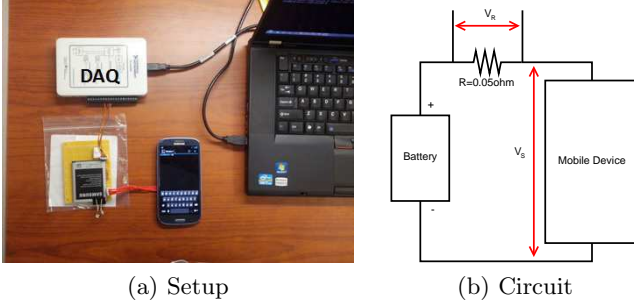


Figure 1: Setup for Energy Profiling

the development of energy models for other mobile devices as future work.

### 3.1 Setup

Our setup consists of a wired server (a desktop equipped with an Intel Quad Core I7-3770 CPU and 32 GB of memory) connected through a single Gigabit Ethernet interface to our campus network, a WiFi access point (IEEE 802.11g), and a mobile device (Samsung Galaxy S3 for AT&T, SGH-I747).

The server is connected through a single Gigabit Ethernet interface to our campus network and runs Ubuntu Linux 12.04 with the MPTCP implementation [17]. The mobile device is connected to the Internet using both 3G(HSDPA) or 4G(LTE) from AT&T and WiFi. The Linux MPTCP kernel is ported into SGH-I747 running a customized Jellybean 4.1.2 platform using a 3.0.2. kernel [9].

To measure the energy consumption of the mobile device, we build an electric circuit with an externally installed battery and a  $R = 0.05\Omega$  high precision resistor between the battery and the mobile device as shown in Figure 1. We use a National Instruments Data Acquisition system (NI-DAQ), with sampling rate of 100K samples/s, in order to measure the voltage supplied to the device ( $V_S$ ) and the voltage drop across the resistor ( $V_R$ ). The energy consumption at each measurement point ( $P$ ) is calculated based on  $R$ ,  $V_R$  and  $V_S$  as  $P = \frac{V_R}{R} \times V_S$ . Energy consumption traces are collected during ten repeated uploads/downloads of files with size varying from 16KB to 16MB. While profiling TCP energy consumption, we force the mobile device CPU to remain in performance mode so as to avoid energy consumption changes due to the CPU switching power modes.

### 3.2 Profiling TCP Energy Consumption over a Single Interface

#### 3.2.1 Promotion and Tail States

Table 1 presents the average duration and energy consumption of the *promotion* and *tail* states measured

Table 1: Summary on Promotion and Tail States (Standard Deviations in Parentheses)

State		Average Duration (sec)	Average Energy Consumption (J)	Fixed Energy Overhead (J)
HSDPA	Promotion	2.098 ( $\pm 0.455$ )	1.463 ( $\pm 0.306$ )	11.337
	Tail	16.123 ( $\pm 1.137$ )	9.873 ( $\pm 1.057$ )	
LTE	Promotion	0.405 ( $\pm 0.047$ )	0.311 ( $\pm 0.041$ )	2.908
	Tail	11.490 ( $\pm 0.492$ )	2.597 ( $\pm 0.275$ )	
WiFi	Promotion	0.095 ( $\pm 0.029$ )	0.040 ( $\pm 0.017$ )	0.149
	Tail	0.295 ( $\pm 0.152$ )	0.109 ( $\pm 0.080$ )	

Table 2: Coefficients for Packet Transfer State

State		Interface		
		HSDPA	LTE	WiFi
Download	$\alpha^d$	9.3440	10.0427	4.6750
	$\beta^d$	-0.9286	-0.8910	-0.8179
Upload	$\alpha^u$	12.5294	13.3438	3.6135
	$\beta^u$	-0.8524	-0.8358	-0.6617

from the collected traces. The energy overheads due to the *promotion* and *tail* are constant for every single packet transfer starting from a low power state. We refer to such overheads of each interface as the *fixed* energy overhead. WiFi has a comparatively short *promotion* and *tail* state, resulting in smaller fixed overheads compared with 3G (HSDPA) and 4G (LTE).

While the HSDPA and LTE *tail* periods are similar, in the range of 12~16 seconds, the HSDPA *promotion* period is around two seconds, roughly five times longer than that of LTE. Since both the *promotion* and *tail* states of WiFi are much shorter, and the per-second energy cost is significantly less, the fixed overhead for WiFi is much lower than for the others.

#### 3.2.2 Packet Transfer State

We use a simple energy model [1, 8] with available bandwidth as the input parameter to model energy consumption during the packet transfer state. We assume that the energy consumption per transferred byte can be represented as a function of the upload/download TCP throughput. To estimate this function for each interface, we explore the energy consumption per transferred byte during packet transfer.

Figures 2 and 3 illustrate the measured energy consumption per transferred byte as a function of obtained TCP throughputs. We observe that the regression model  $P = \alpha \times B^\beta$  yields estimates of  $P$  close to actual measured values as a function of  $B$ , the available throughput. Table 2 lists the estimates of  $\alpha$  and  $\beta$  for each setting. For example, given a download throughput over the LTE interface of  $B_L$  (Mbps), the energy consumption for downloading each byte,  $P_L$  ( $\mu J/B$ ), is defined

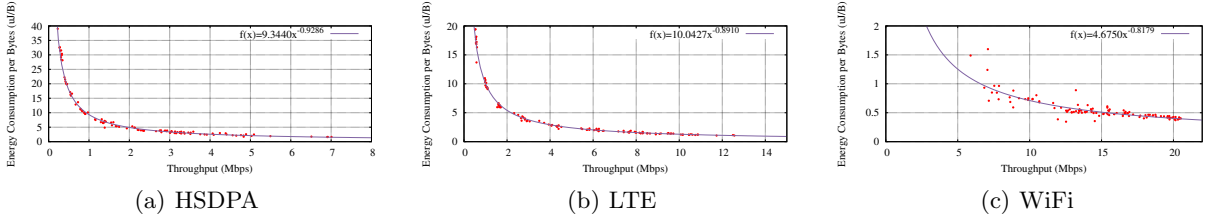


Figure 2: Energy Consumption per Byte during Packet Transfer - Downloads

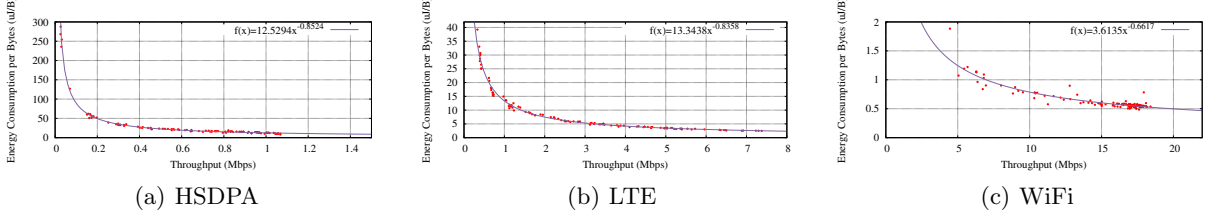


Figure 3: Energy Consumption per Byte during Packet Transfer - Uploads

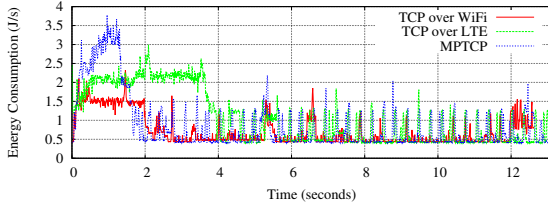


Figure 4: Sample Energy Trace - 4MB Download

as:

$$P_L(B_L) = \alpha_L \times B_L^{\beta_L},$$

where  $\alpha_L = 10.04$  and  $\beta_L = -0.89$ . Here, we do not separately consider the energy consumption due to sending TCP ACK packets due to their small impact: since the size of the ACK packet is comparatively small, we consider the consumed energy for sending an ACK as part of the overall energy consumption for receiving data packets.

## 4. MPTCP ENERGY MODEL

Here we present our MPTCP energy model for the case that MPTCP simultaneously uses the LTE and WiFi interfaces based on the profiling results in the previous section.

### 4.1 Model

Figure 4 shows sample energy trace for TCP over each interface and MPTCP when the device downloads a 4MB file. We first observe that MPTCP consumes more energy during the first 2 seconds of data transfer. We also see that both TCP over LTE and MPTCP incur energy costs for the LTE *tail* state (TCP over LTE starts its tail at around 4 seconds, and MPTCP starts it at around 2 seconds). Analogous to the en-

ergy model of standard TCP over each interface [8], we model MPTCP energy consumption as a function of the available throughput of each interface.

Let  $B_L$  and  $B_W$  denote the throughputs of LTE and WiFi, respectively. Suppose that  $S$  is the size of the file and  $S_W$  and  $S_L$  are the number of bytes transferred through WiFi and LTE, respectively;  $S = S_W + S_L$ . A simple estimate of MPTCP energy consumption is to sum the energy consumed by each interface over transferred packets. However, we observe that the MPTCP energy consumption is slightly less than this, possibly due to the shared use of energy. To consider such shared energy consumption, we assume that a device consumes a fraction  $\gamma$  of the sum during the overlapped period for packet transfer. In the case of the fixed energy overhead for *promotion* and *tail*, we assume that the overhead is separately consumed for each interface. Let  $\theta$  denote the ratio of the duration of the data transfer when both interfaces are simultaneously transferring packets. Given  $S_W$  and  $S_L$ , we approximate  $\theta$  as:

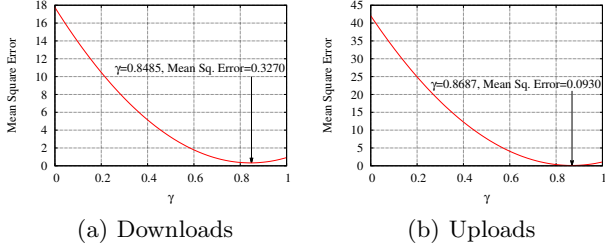
$$\theta = \frac{\min(S_W/B_W, S_L/B_L)}{\max(S_W/B_W, S_L/B_L)},$$

Based on our assumption, we estimate the MPTCP energy consumption during packet transfer,  $E_T$ , as:

$$E_T = (P_W(B_W) \times S_W + P_L(B_L) \times S_L) (1 - \theta + \gamma\theta),$$

where  $B_W$  and  $B_L$  are the available throughputs over WiFi and LTE and  $S_W$  and  $S_L$  are transferred bytes over WiFi and LTE, respectively. The total MPTCP energy consumption, including the energy overheads associated with the *promotion* and *tail* states of each interface, are then represented as:

$$E_M = E_T + C_W + C_L,$$



**Figure 5:  $\gamma$  that minimizes mean square error**

where  $C_W$  and  $C_L$  are the fixed energy overheads for the *promotion* and *tail* states of the WiFi and LTE interfaces, respectively.

## 4.2 Determination of $\gamma$

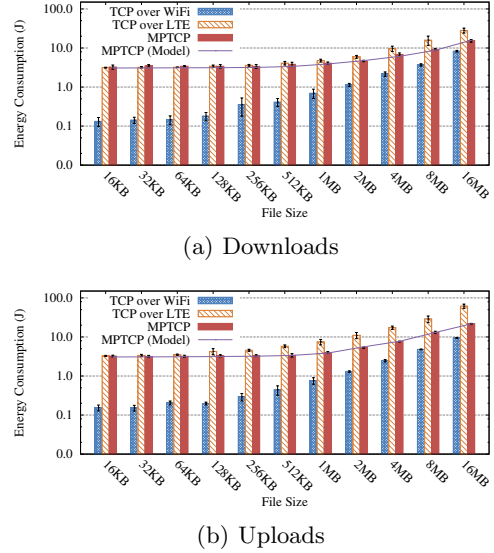
To determine  $\gamma$ , we perform a set of experiments to measure MPTCP energy consumption while a device downloads or uploads files of various sizes using MPTCP with WiFi and LTE. We choose  $\gamma$  to minimize the mean square error between measured and estimated values.

Figure 5 shows the mean square error between measured and estimated energy consumption as a function of  $\gamma$  when MPTCP uses WiFi and LTE. As shown in Figure 5, the mean square errors are minimized when  $\gamma$  is equal to 0.8485 (for downloads) and 0.8687 (for uploads). Approximately 13 ~ 16% of energy appears to be consumed by shared components when MPTCP is simultaneously operating over both the WiFi and LTE interfaces. One example of shared energy consumption might be CPU processing power to handle MPTCP operations.

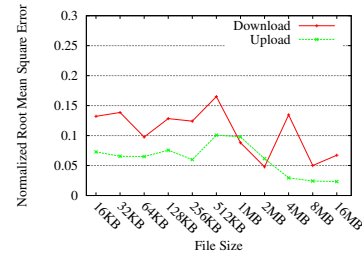
In addition, our phone uses the Qualcomm Snapdragon S4 [19, 21] with the LTE integrated on the chip, thus, it seems likely that they share some amount of energy while operating. Since we do not have enough knowledge on the electrical schematic and operation of the mobile device, in the rest of this paper, we use the measured values  $\gamma^d = 0.8485$  (for downloads) and  $\gamma^u = 0.8687$  (for uploads) for our MPTCP energy model. Note that in these experiments, the device uses WiFi and LTE interfaces for downloading and uploading:  $\gamma$  can differ in the case when MPTCP uses WiFi and HSDPA interfaces. Also, as with the profiling results in the previous section, using different mobile devices may result in different estimates for  $\gamma$ . We will explore appropriate values of  $\gamma$  for various mobile devices in future work.

## 4.3 Model Validation

Figure 6 shows the average total energy consumption of TCP over each interface (WiFi and LTE) and MPTCP when our device downloads/uploads a file of fixed size ten times each for several different file sizes.

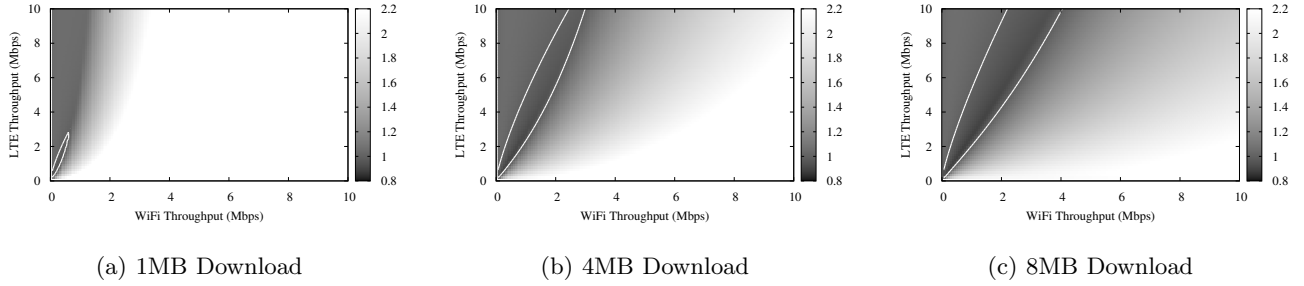


**Figure 6: Total Energy Consumption according to File Size**



**Figure 7: Normalized RMSE**

We observe that MPTCP is less energy efficient than single-path TCP over WiFi as the file size decreases. This is because of the fixed overhead from the *promotion* and *tail* state of LTE: when a file size is small, the device spends energy to establish an LTE subflow connection even though it rarely transmits packets through LTE. In our experiments, MPTCP does not even utilize an LTE subflow until the transfer size becomes larger than 512KB (effective use of LTE starts even after a transfer becomes larger than 2MB). This is due to the relatively large RTT (about 65 ms for downloads and 95 ms for uploads) over the LTE path; setting up a subflow over LTE takes longer than over WiFi, which has much lower RTT (about 15 ms for downloads and uploads). The larger RTT also means it takes time for the congestion window of the LTE subflow to open up [3]. Thus, a small file (<512KB) transmission completes in a short time and consumes a relatively small amount of energy for packet transfer compared to the fixed overhead. Consequently, MPTCP yields similar energy consumption to TCP over LTE when a file is smaller than 512KB, which is larger than that of TCP over WiFi, even though it allocates almost of all traffic



**Figure 8: MPTCP Total Energy Consumption Normalized by Most Energy Efficient TCP**

to WiFi. We see that in the case of small file transmissions, MPTCP uses power inefficiently, whereas TCP over WiFi is more energy efficient.

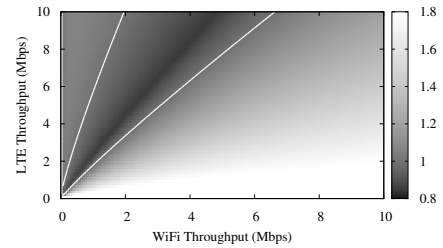
We also calculate the expected MPTCP energy consumption based on our model. As shown in Figure 6, our model accurately estimates the energy consumption of MPTCP. Figure 7 presents the root mean square errors (RMSE) normalized by the average measured energy consumption for downloading/uploading files of each size. As shown in Figure 7, the MPTCP energy model estimates differ from the measure values by less than 17%. As file size increases, the MPTCP energy model becomes more accurate. Note that the large percentage error when the file size is small ( $<1\text{MB}$ ) is because the entire energy consumption is relatively small ( $<5\text{J}$ ), even though an absolute error is small (error of  $<1\text{J}$ ).

## 5. ENERGY AWARE MPTCP

In this section, we introduce eMPTCP which improves on standard MPTCP by being more energy efficient. We focus on downloads over the WiFi and LTE interfaces since they are more common.

### 5.1 Motivation

First, based on our MPTCP energy model, we characterize the throughput region where MPTCP is more energy efficient than standard TCP and MPTCP, given the file size for downloads. For example, Figure 8 presents the MPTCP energy consumption estimated for 1MB, 4MB, and 8MB downloads, normalized by the energy consumption of standard TCP over the most energy efficient interface given the achieved throughput over each interface. In Figure 8, the regions inside white curves correspond to the throughput values where MPTCP is more energy efficient than either TCP over WiFi or TCP over LTE. At the left side of the region, TCP over LTE is the most energy efficient while TCP over WiFi is the most energy efficient at the right of the region. We have also observed that the operating region where MPTCP is most energy efficient becomes smaller as file



**Figure 9: Energy Consumption per Downloaded Byte of Using WiFi and LTE Normalized by Most Energy Efficient TCP**

size decreases. When file sizes are smaller than 1MB, MPTCP is more efficient than standard TCP only in a significantly limited region and TCP over WiFi is better than MPTCP even when the available WiFi throughput is small. Inspired by this finding, in Section 5.2, we propose a delayed LTE subflow establishment mechanism that allows MPTCP to reduce unnecessary energy consumption due to the fixed overhead while downloading a small file.

We also investigate the energy efficiency of using both WiFi and LTE interfaces according to available throughput during packet transfer. Figure 9 presents the energy consumption per downloaded byte over both WiFi and LTE, normalized by that achieved using the best single interface. As with Figure 8, using both interfaces consumes the smallest amount of energy to download a byte inside the white curves. This shows that we should be careful to choose the appropriate combination of interfaces in order to save energy according to available throughput of each interface. Based on this result, we propose a subflow usage management algorithm in Section 5.3.

### 5.2 Delayed LTE Subflow Establishment

The delayed subflow management algorithm is shown in Algorithm 1 with notation defined in Table 3. Algorithm 1 is executed only when eMPTCP initiates a new connection. We assume that a device uses the WiFi in-

**Table 3: Notations**

Symbol	Definition
$\kappa$	Download Amount Threshold for delaying LTE subflow establishment
$\tau$	Timer Threshold for delaying LTE subflow establishment (ms)
$\delta$	Bandwidth Estimation Interval (ms)
$h$	Holt-Winters step ahead parameter
$\rho$	Holt-Winters EWMA parameter

**Algorithm 1** Delayed LTE subflow Establishment

---

```

if MPTCP starts a new connection with WiFi then
  Postpone LTE subflow establishment
  Trigger a timer that expires in  $\tau$  ms
else
  Establish all remaining subflows
end if
while delaying LTE subflow establishment do
  if Download more than  $\kappa$  bytes or Timer is expired then
    Establish LTE subflow
  end if
end while

```

---

interface as the primary network interface. In this case, the device tries to set up an LTE subflow only some time after establishing an WiFi subflow. When the file to be downloaded is small, we want to avoid the unnecessary expenditure of energy to establish an LTE subflow connection. To avoid such an unnecessary LTE subflow establishment, eMPTCP introduces a delay between WiFi and LTE subflow establishment: eMPTCP does not start the LTE subflow until it receives  $\kappa$  bytes through the WiFi interface. Using the delayed LTE subflow establishment, eMPTCP does not consume energy for the *promotion* and *tail* states of LTE interface when a device downloads a file smaller than  $\kappa$  bytes. However, if the available WiFi throughput is extremely small, using only the WiFi subflow until the threshold  $\kappa$  can incur more energy consumption than using both (see Figure 8). For example, if the WiFi bandwidth is smaller than 3Mbps and the LTE bandwidth is larger than 10Mbps when downloading a 8MB file, using both is better. Therefore, to prevent using only a slow WiFi subflow, eMPTCP uses a timer to trigger LTE subflow establishment: if the timer expires after  $\tau$  seconds, eMPTCP establishes an LTE subflow even though the downloaded amount through WiFi does not reach the threshold  $\kappa$ .

### 5.3 Subflow Usage Management

After establishing an MPTCP connection, eMPTCP uses the subflow management algorithm shown in Algorithm 3. The algorithm decides whether to use both interfaces or WiFi-only for data transfer based on the estimated available throughput of each interface. Note that eMPTCP does not switch to using LTE-only since its expected gain is not much more than using both interfaces, as shown in Figure 9. Thus, the right white

**Algorithm 2** Holt-Winters Bandwidth Predictor

---

```

function HOLT-WINTERS_BW_PREDICTOR( $Y_i, h$ )
  if  $i \leq 2$  then
     $a \leftarrow Y_i$ 
     $b \leftarrow Y_{i-1}$ 
  else
     $temp \leftarrow a$ 
     $a \leftarrow \rho \times Y_i + (1 - \rho)(a + b)$ 
     $b \leftarrow \rho \times (a - temp) + (1 - \rho)b$ 
  end if
  Return  $a + bh$  /* return h-step ahead prediction */
end function

```

---

**Algorithm 3** Subflow Management

---

```

for every  $\delta$  ms do
   $i \leftarrow i + 1$ 
   $Y_i = B/\delta$  /*  $B$ : Downloaded bytes for  $\delta$  ms via subflow */
   $BW = \text{Holt-Winters\_BW\_Predictor}(Y_i, h)$ 
  if subflow is associated with WiFi interface then
     $WiFi\_BW = BW$ 
  else
     $LTE\_BW = BW$ 
  end if
  if  $WiFi\_BW$  and  $LTE\_BW$  in WiFi-only region then
    Suspend LTE subflow
  else
    Resume LTE subflow
  end if
end for

```

---

curve in Figure 9 determines the throughput thresholds for eMPTCP to switch between using both and WiFi-only. To estimate current available throughput, eMPTCP samples downloaded bytes through each interface every  $\delta$  ms. Given the sampled throughputs, to predict bandwidth changes, eMPTCP uses a Holt-Winters time-series forecasting algorithm [18], which is known as a more accurate predictor than formula-based predictors using a function of underlying path characteristics [7]. The Holt-Winters algorithm is shown in Algorithm 2.

When a device decides to switch from using both interfaces to WiFi-only or vice versa, this decision needs to be communicated to the sender-side. To inform the sender side of the state change, we add an MP\_PRIO option [5], which changes the priority of the LTE subflow, to the next packet to be transmitted.

When the sender needs to switch from using WiFi-only to using both interfaces, eMPTCP at the sender needs utilize the LTE subflow quickly. To this end, eMPTCP disables CWND reset after an idle period longer than the retransmission timeout in RFC2861 [6] to ensure that an LTE subflow avoids unnecessary slow-start when eMPTCP starts re-using the LTE subflow. Also, eMPTCP sets the measured round trip time (RTT) of the LTE subflow to zero when it releases the low priority status of LTE subflow. This modification enables an LTE subflow to be quickly probed by the MPTCP subflow scheduler, since it selects a subflow with the lowest RTT for packet transmission [17].



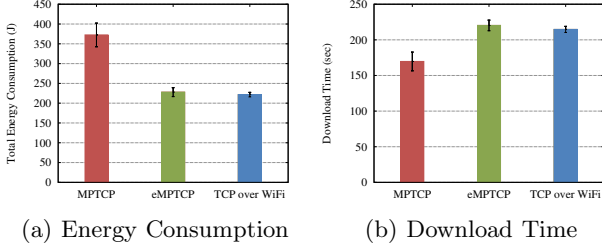


Figure 10: Persistent High WiFi Bandwidth

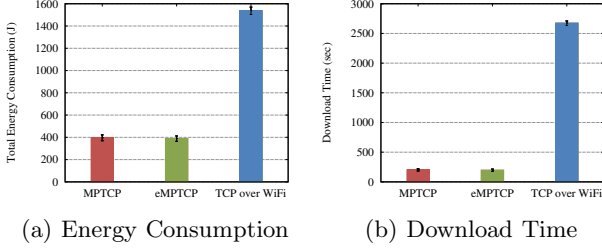


Figure 11: Persistent Low WiFi Bandwidth

## 6. EVALUATION

In this section, we evaluate eMPTCP in terms of energy consumption and download performance, comparing it to standard MPTCP and TCP over WiFi. We consider three experimental scenarios where we change the WiFi bandwidth, the background traffic, and the presence of mobility, which are described in more detail below.

### 6.1 Setup

We deploy the same setup used in our single-path TCP profiling measurements in Section 3.1. We set eMPTCP parameters as follows: the download amount threshold  $\kappa$  to postpone an LTE subflow is set to 1MB since MPTCP is rarely more energy efficient than single path TCP when downloading a file smaller than 1MB as shown in Figure 8(a). The timer threshold  $\tau$  is set to 3 seconds. We set the polling interval  $\delta$  to sample throughput every 200 ms, which yields the best energy performance in our preliminary experiments with different values of  $\delta$ . eMPTCP uses a one step ahead predicted value with  $\rho = 0.125$  from the Holt-Winters forecasting algorithm to decide whether to use both interfaces or WiFi-only.

### 6.2 Experiments with Static Configuration

The purpose of these experiments is to show that, in relatively simple static environments, eMPTCP makes the proper path decisions to optimize power consumption. We measure energy consumption and download time of eMPTCP, MPTCP, and TCP over WiFi for two extreme cases: persistent high ( $>10\text{Mbps}$ ) and low ( $<1\text{Mbps}$ ) WiFi bandwidth while the device downloads

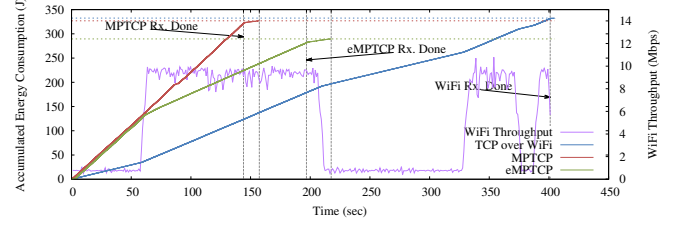


Figure 12: Accumulated Energy Consumption Example with Random WiFi Bandwidth Change

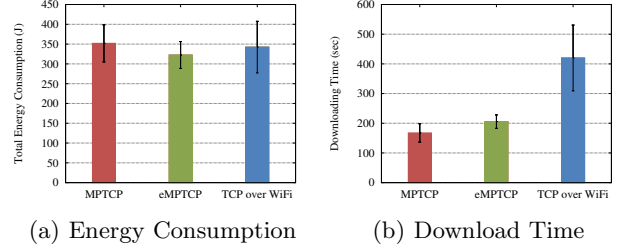


Figure 13: Random WiFi Bandwidth Changes

a 256MB file at a static location. Figures 10 and 11 compare the energy consumption and download times averaged over five runs for the two cases. In the first case, WiFi bandwidth is high, and thus using it is more power efficient than using LTE or both interfaces. Figure 10 shows that eMPTCP chooses WiFi-only, effectively behaving like TCP over WiFi. In contrast, when the WiFi bandwidth is small ( $<1\text{Mbps}$ ), and thus less energy-efficient as LTE, Figure 11 shows that eMPTCP yields almost the same performance as MPTCP by using both interfaces (after the LTE startup delay determined by parameters  $\kappa$  and  $\tau$ ). This illustrates that eMPTCP seeks the most energy efficient path usage, while preserving MPTCP's robustness to path degradation, and without user involvement.

### 6.3 Experiments with Bandwidth Changes

The goal of these experiments is to examine how robust eMPTCP is to changes in network bandwidth, which can exhibit variability. In these experiments, link bandwidth changes while the device downloads a 256MB file at a static location. To simulate available throughput changes, since we do not have any control of the LTE network, we only change the available bandwidth of WiFi by randomly setting the 802.11 physical layer bit rate of our AP between 1Mbps and 18Mbps, where the achieved TCP bandwidth are  $<1\text{Mbps}$  and  $>10\text{Mbps}$ , respectively.

Here, we explore scenarios with random WiFi bandwidth changes where the time between WiFi bandwidth changes is exponentially distributed with a mean of 40 seconds. Figure 12 presents an example trace of accu-



ulated energy consumption of eMPTCP, MPTCP, and TCP over WiFi with random WiFi bandwidth changes. At the beginning, eMPTCP uses both interfaces after the LTE startup delay since WiFi throughput is too low to be energy efficient. However, eMPTCP suspends the LTE subflow after WiFi bandwidth increases while MPTCP continues to use both interfaces. By suspending the LTE subflow, through which more energy cost to complete the download is expected even with additional bandwidth, eMPTCP spends less energy but completes the download later than MPTCP. Compared with TCP over WiFi, eMPTCP completes the download sooner and consumes less energy.

Figure 13 compares energy consumption and download time averaged over ten runs. In this scenario, eMPTCP consumes approximately 8% and 6% less energy than MPTCP and TCP over WiFi, respectively. However, eMPTCP is approximately 22% slower than MPTCP since it utilizes an LTE subflow only when an LTE subflow can provide an energy gain with additional throughput. In contrast, by utilizing an LTE subflow when energy gain is available (WiFi throughput is  $< 1\text{Mbps}$ ), eMPTCP completes downloads twice as fast as TCP over WiFi and consumes less energy. This shows that eMPTCP offers the robustness of MPTCP while obtaining greater energy efficiency than MPTCP. Note that the performance gain achieved by eMPTCP differs according to how WiFi bandwidth changes. If WiFi bandwidth is frequently changing, the switching overhead in eMPTCP may become noticeable: an LTE interface triggers another *promotion* and *tail* states when it starts to be used again.

## 6.4 Experiments with Background Traffic

It is well known that multiple WiFi nodes can contend for the air channel, causing interference and loss (e.g., [11]). The goal of this next section is to see how well eMPTCP copes with with random background traffic. Background traffic causes available throughput changes similar to link bandwidth changes, but also results in contention and interference in the communication channel. In these experiments, we utilize two or three interfering nodes, which use the same WiFi channel as the mobile device. While the device downloads a 256MB file at a static location, each node turns UDP traffic on and off for a random duration, which is exponentially distributed with a rate of  $\lambda_{\text{on}}$  and  $\lambda_{\text{off}}$ . We fix  $\lambda_{\text{on}} = 0.05$ , and then perform experiments with  $\lambda_{\text{off}} = 0.025$  and  $\lambda_{\text{off}} = 0.05$ . As with the previous scenario, we control background traffic for the WiFi channel only.

Figure 14 presents the average round trip time (RTT) of TCP over WiFi, which is one indicator of the interference level in the WiFi channel, with different numbers of interfering nodes ( $n$ ) and different  $\lambda_{\text{off}}$ . While larger  $n$  apparently increases the average WiFi RTT,  $\lambda_{\text{off}}$  does

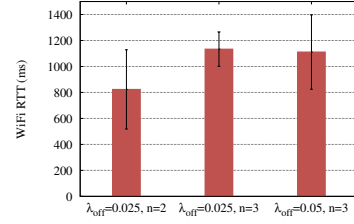
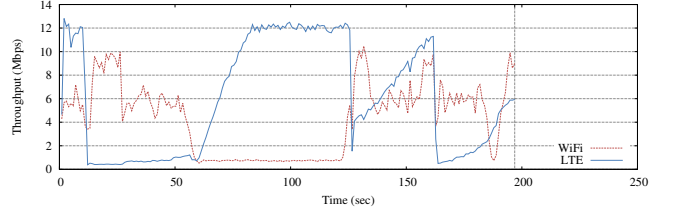
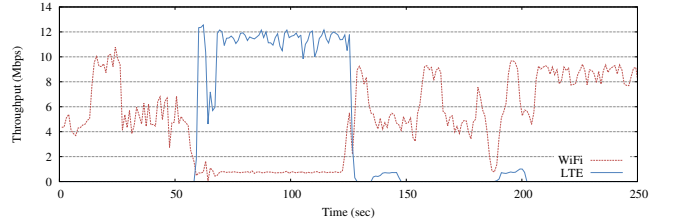


Figure 14: Round Trip Time of TCP over WiFi according to Background Traffic Setup



(a) MPTCP



(b) eMPTCP

Figure 15: Example Throughput Trace with Random WiFi Background Traffic ( $n = 2$ ,  $\lambda_{\text{on}} = 0.05$ , and  $\lambda_{\text{off}} = 0.025$ )

not significantly affect the average RTT (when comparing the cases of  $\lambda_{\text{off}} = 0.025$  and  $\lambda_{\text{off}} = 0.05$  with  $n = 3$ ). This is because 20 more seconds per individual period without background traffic ( $1/0.025 - 1/0.05$ ) might not be long enough to change the average RTT. However, we can expect the device to obtain more WiFi throughput as background traffic periods become longer (with smaller  $\lambda_{\text{off}}$ ).

Figure 15 shows example throughput traces of MPTCP and eMPTCP when two interfering nodes turn on and off traffic with  $\lambda_{\text{on}} = 0.05$  and  $\lambda_{\text{off}} = 0.025$ . We observe that MPTCP itself is likely to avoid too aggressive use of an LTE subflow while a WiFi subflow can provides high bandwidth, e.g., at around time 10~60 seconds in Figure 15(a). This is because the MPTCP subflow scheduler chooses a subflow for packet transmission that has sufficient CWND and the smallest RTT [17]. However, MPTCP consumes energy utilizing an LTE subflow even with such a small throughput gain. In contrast, eMPTCP suspends an LTE subflow while WiFi bandwidth is sufficiently large in order to avoid energy

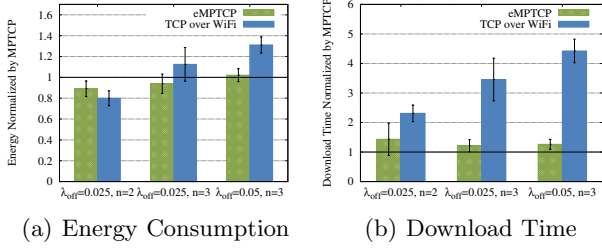


Figure 16: Random WiFi Background Traffic

inefficient path usage. In Figure 15(b), we note that eMPTCP incorrectly determines to use an LTE subflow around 140~150 and 190~200 seconds. This is caused by the sudden step WiFi throughput decreases at 140 and 190 seconds, which result in incorrect throughput predictions of eMPTCP. However, eMPTCP stops using the LTE subflow after obtaining improved throughput estimates.

Figure 16 presents the average energy consumption and download times for different values of  $n$  and  $\lambda_{off}$ , normalized relative to MPTCP, i.e., smaller numbers are better for both (a) and (b), and when lower than one are better than MPTCP. Values are averaged over five experiments.

As shown in Figures 16(a), eMPTCP consumes less energy than MPTCP as  $n$  and  $\lambda_{off}$  become smaller. Comparing the cases of  $n = 2$  and  $n = 3$  with  $\lambda_{off} = 0.025$ , we observe that the energy efficiencies of eMPTCP and TCP over WiFi are larger with  $n = 2$ . Recall that the energy efficiency of eMPTCP is improved when it can suspend an LTE subflow in situations where using WiFi only is more efficient. Larger numbers of interfering WiFi nodes result in more losses caused by collisions when background traffic is present, resulting in more CWND decreases. Thus, the device is likely to obtain more TCP bandwidth with a larger CWND when background traffic disappears, resulting in better energy efficiency of TCP over WiFi and eMPTCP. Note that TCP over WiFi is most energy efficient when  $n$  and  $\lambda_{off}$  are equal to 2 and 0.025, respectively. However, as shown in Figure 16(b), in that setting, TCP over WiFi requires twice as much time to complete downloads as eMPTCP, while it consumes just 11% less energy. We see that eMPTCP is more efficient than TCP over WiFi in terms of the tradeoff between energy consumption and performance.

As shown in Figure 16(b), MPTCP provides the best download times, regardless of the values of  $n$  and  $\lambda_{off}$ , since it always utilizes an LTE subflow. Also, as we expect, the download time of TCP over WiFi becomes significantly larger as  $n$  and  $\lambda_{off}$  increase. Compared with MPTCP, eMPTCP requires 20~40% more time while consuming 9~11% less energy: eMPTCP cannot achieve energy gain as much as the amount of perfor-

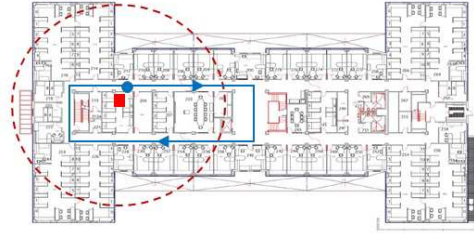


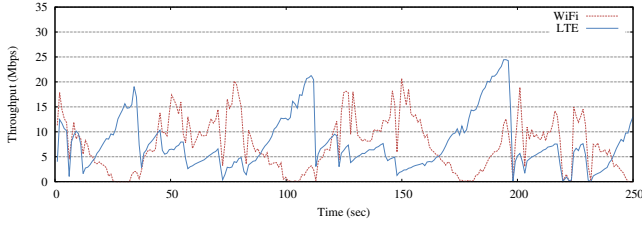
Figure 17: Mobile Scenario inside the UMass CS Building. Route starts at the blue point. The red square is the AP. The red dashed circle is the estimated usable access range of the AP.

mance degradation. This might be because eMPTCP may sometimes poorly predict bandwidth due to fluctuating throughputs, as shown in Figure 15(b), and it also has additional energy overhead to suspend and resume an LTE subflow due to the *promotion* and *tail* state. In our experiments, the average lengths of period during WiFi background traffic turns off (an LTE subflow is supposed to be suspended) are 20 and 40 seconds. These are not long enough to allow one to ignore energy overhead for *tail* state of LTE that lasts for around 12 seconds. Compared to TCP over WiFi, eMPTCP obtains a significant performance improvement (up to 70% less) in download time, while at the same time using less energy.

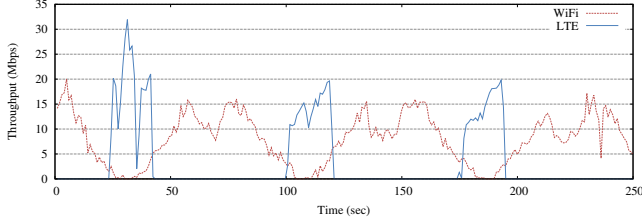
## 6.5 Experiments with Mobility

Finally, mobile devices do not simply use wireless networks, they can also move around their environments, causing connectivity changes. The goal of the experiments in this section is to determine how well eMPTCP performs and adapts in an actual mobile scenario. We measure energy consumption and download amount while moving for 250 seconds along the route shown in Figure 17. To make our comparison between MPTCP and eMPTCP as fair as possible, we use as similar routes as possible for the experiments.

Figures 18 and 19 present example traces of throughput and accumulated energy consumption from our mobile scenario. In this experiment, the device is sometimes within the WiFi communication range, and sometimes outside it, depending on its location. As the device moves outside the WiFi communication range, WiFi throughput decreases, e.g., the duration around 25~40 seconds in Figure 18(a) and (b). At the beginning of the route, MPTCP starts by establishing both subflows, whereas eMPTCP postpones establishing an LTE subflow, since WiFi throughput is high enough to be more energy efficient than using both interfaces. However, eMPTCP establishes an LTE subflow after the WiFi bandwidth decreases when the device is leav-

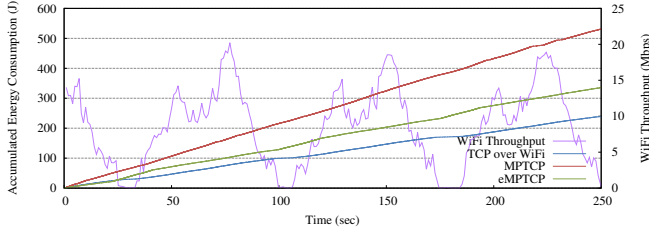


(a) MPTCP



(b) eMPTCP

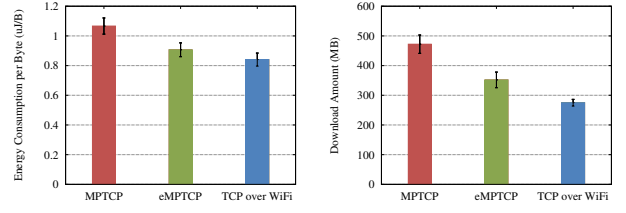
**Figure 18: Example Throughput Trace with Mobile Scenario**



**Figure 19: Example of Accumulated Energy Consumption with Mobile Scenario**

ing the WiFi communication range (at around 25 seconds in Figure 18(b)). Then, whenever the device cannot obtain enough WiFi bandwidth to be more energy efficient than using both interfaces, eMPTCP utilizes the LTE subflow rather than only using the bad WiFi subflow. In this experiment, we observe that since the device is inside WiFi communication range most of time, eMPTCP utilizes the LTE subflow only for a few short periods. Therefore, as shown in Figure 19, the slope of eMPTCP’s accumulated energy consumption (the energy consumption per second) is larger than that of the slope of TCP over WiFi, but smaller than that of MPTCP.

We now examine the per-byte energy efficiencies of MPTCP, eMPTCP, and TCP over WiFi. Figure 20 compares the energy consumption per byte and download amount for 250 seconds averaged over five runs. Since eMPTCP utilizes the LTE subflow for only several short periods, its energy consumption per byte is 8% larger than that of TCP over WiFi and 15% smaller than that of MPTCP. Because WiFi throughput degradation is due only to the distance between the AP and



(a) Energy Consumption per Byte

(b) Download Amount

**Figure 20: Mobile Scenario**

device (since there is no WiFi background traffic in these experiments), TCP over WiFi is slightly better in terms of energy efficiency than eMPTCP, similar to the case of  $n = 2$  and  $\lambda_{\text{off}} = 0.025$  in Figure 16(a).

Comparing the download amounts during the experiments, we observe that eMPTCP downloads 28% more data than TCP over WiFi even though it yields a similar per-byte energy efficiency to TCP (just 8% more energy consumption per byte). As was the case in the experiments with random WiFi background traffic, this result shows that eMPTCP can obtain significant performance gains in the tradeoff between energy consumption and performance, compared to TCP over WiFi. Recall that eMPTCP consumes 15% less energy per byte than MPTCP. With such better per-byte energy efficiency, eMPTCP downloads 25% less data. Similar to the results of Figure 16, eMPTCP still loses proportionally more in performance than it saves in energy efficiency, due to the overhead of switching between WiFi-only and using both interfaces. Improving eMPTCP to achieve better efficiency by considering switching overhead is future work.

## 6.6 eMPTCP vs. MPTCP with WiFi First

Raiciu et. al [16] propose a simple strategy called MPTCP with WiFi First, where MPTCP only uses WiFi when available, and only uses the cellular network otherwise. This is accomplished by placing the cellular subflow in backup mode and activating it only when WiFi is not available. This simple strategy may seem similar to eMPTCP, however, it cannot take advantage of more dynamic situations where TCP over LTE or MPTCP is more energy-efficient than TCP over WiFi. MPTCP with WiFi First can only utilize an LTE subflow when the WiFi subflow explicitly breaks, such as due to a WiFi AP disassociation. For example, in our mobile scenario in Section 6.5, MPTCP with WiFi First would not use the LTE subflow even when the WiFi subflow becomes unusable, e.g., the duration around 25~40 seconds in Figure 18(a) and (b), since the device does not lose the WiFi association. Therefore, if WiFi provides too low bandwidth to be more energy efficient than LTE while it is still associated, MPTCP

with WiFi First degenerates into single-path TCP over WiFi, which yields inefficient energy usage as shown in the previous subsections. In contrast, eMPTCP suspends and resumes an LTE subflow according to the estimated throughputs and energy efficiencies of each interface, regardless of the state of the association. Thus, eMPTCP can adaptively control subflow usage so as to more quickly respond to available bandwidth changes and obtain more energy efficiency than MPTCP with WiFi First.

## 7. RELATED WORK

Balasubramanian et al. [1] measures energy consumption on a Nokia N95 platform and identifies high energy overhead as the result of the tail state in 3GPP interfaces (GSM, 3G). They develop a protocol called *tailender* to schedule transfers so as to minimize energy consumed by the tail.

Huang et al. [8] provide an in-depth look at power and performance characteristics of 4G LTE networks based on a large-scale measurement of a commercial cellular provider. They show that while LTE is more energy-efficient than 3G, it is still not as efficient as WiFi, partially due to the tail state cost in LTE.

Schulman et al. [20] show that the power consumed by wireless radios is higher when the signal is weak. They present Bartendr, a system for scheduling transmissions when the signal is strong so as to minimize power consumption. Ding et al. [4] provide a more in-depth analysis of the impact of signal strength.

Rahmati et al. [15] also examine how to reduce power consumption by utilizing the most efficient interface. Noting that WiFi is much more efficient than 3G, they devise algorithms to estimate WiFi conditions without powering up the antenna, showing a 39% improvement via simulation in energy consumption.

Ra et al. [14] show how a delay-tolerant application can postpone its network usage until a more energy-efficient device is available, saving 10~40% of energy.

Bui et al. [2] implement a middleware to aggregate bandwidth of asymmetric wireless links for video streaming, called GreenBag. GreenBag estimates the available bandwidth of WiFi and LTE and determines the amount of allocated traffic to each interface while considering a quality-of-service requirement of streaming service. The authors show that GreenBag reduces energy consumption by 14~25% compared with a bandwidth aggregation for throughput maximization. However, since GreenBag operates above TCP layer as a system background process, it only works for modified HTTP requests sent not to original destinations but to GreenBag, therefore, each application needs to be modified to cooperate with GreenBag.

The most closely related work to ours is Pluntke et al. [13] who introduce the concept of scheduling paths

in MPTCP to minimize energy consumption using a scheduler based on Markov decision processes. The path scheduling decision is made periodically every  $t$  time units. Schedulers are computed in the cloud and downloaded periodically to the device. They evaluate their scheduler via simulation, by using models of device energy consumption and find they can reduce energy consumption by almost 10 % in one scenario (out of four), viewing high-quality video stream. Our work, in contrast, is measured and evaluated experimentally using a real MPTCP implementation on a physical device.

Raiciu et al. [16] look at a number of issues in using MPTCP for mobility, including power consumption. They propose and evaluate a simple strategy that periodically samples both paths for 10 seconds and then uses the more energy-efficient path for 100 seconds. Evaluating their strategy via simulation, the approach is more power efficient than an energy-unaware MPTCP implementation, but achieves lower bandwidth.

Paasch et al. [12] studied mobile/WiFi handover performance with MPTCP. The authors investigated the impact of handover on MPTCP connections using different modes such as Full-MPTCP mode (where all potential subflows are used to transmit packets) and Backup mode (where only a subset of subflows are used). They also measure energy consumption on a Nokia N950 smartphone for two download scenarios and find that using WiFi alone is more energy-efficient than base MPTCP.

## 8. CONCLUSION

This paper proposes a method to improve the energy efficiency of MPTCP. We develop eMPTCP, which manages subflows based on expected energy efficiency given available throughputs. We perform experiments with a real mobile device running MPTCP and eMPTCP. Our experimental results show that eMPTCP is able to consume less energy than MPTCP while it still provides the benefits of MPTCP such as robustness. Our results also show that eMPTCP obtains significantly better performance with comparatively small energy overhead or even with less energy consumption than TCP over WiFi. For future work, we will extend the implementation and experiments to reflect further environments, such as other mobile devices, as well as refining the parameter setting of eMPTCP to improve performance.

## Acknowledgement

This research was sponsored by US Army Research laboratory and the UK Ministry of Defence and was accomplished under Agreement No. W911NF-06-3-0001. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the US Army Research Laboratory, the U.S.

Government, the UK Ministry of Defense, or the UK Government. The US and UK Governments are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation hereon.

## 9. REFERENCES

- [1] N. Balasubramanian, A. Balasubramanian, and A. Venkataramani. Energy consumption in mobile phones: A measurement study and implications for network applications. In *Proc. of ACM IMC*, 2009.
- [2] D. H. Bui, K. Lee, S. Oh, I. Shin, H. Shin, H. Woo, and D. Ban. Greenbag: Energy-efficient bandwidth aggregation for real-time streaming in heterogeneous mobile wireless networks. In *Proc. of IEEE RTSS*, 2013.
- [3] Y.-C. Chen, Y.-S. Lim, R. J. Gibbens, E. Nahum, R. Khalili, and D. Towsley. A measurement-based study of multipath TCP performance in wireless networks. In *Proc. of ACM IMC*, Nov 2013.
- [4] N. Ding, D. Wagner, X. Chen, Y. C. Hu, and A. Rice. Characterizing and modeling the impact of wireless signal strength on smartphone battery drain. In *Proc. of ACM SIGMETRICS*, 2013.
- [5] A. Ford, C. Raiciu, M. Handley, and O. Bonaventure. TCP extensions for multipath operation with multiple addresses. RFC 6824 (Experimental), Jan. 2013.
- [6] M. Handley, J. Padhye, and S. Floyd. TCP congestion window validation. *IETF RFC 2861*, 2000.
- [7] Q. He, C. Dovrolis, and M. Ammar. On the predictability of large transfer tcp throughput. In *Proc. of ACM SIGCOMM*, 2005.
- [8] J. Huang, Q. Feng, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck. A close examination of performance and power characteristics of 4G LTE networks. In *Proc. of ACM Mobisys*, 2012.
- [9] Y.-S. Lim. MPTCP for Android. <http://cs.umass.edu/~ylim/mptcp>.
- [10] Y.-S. Lim, Y.-C. Chen, E. M. Nahum, D. Towsley, and R. J. Gibbens. How green is multipath TCP for mobile devices? To appear in ACM AllThingsCellular, 2014.
- [11] J. Manweiler and R. Roy Choudhury. Avoiding the rush hours: WiFi energy management via traffic isolation. In *Proc. of ACM MobiSys*, 2011.
- [12] C. Paasch, G. Detal, F. Duchene, C. Raiciu, and O. Bonaventure. Exploring mobile/wifi handover with multipath TCP. In *Proc. of ACM Cellnet*, 2012.
- [13] C. Pluntke, L. Eggert, and N. Kiukkonen. Saving mobile device energy with multipath TCP. In *Proc. of ACM MobiArch*, 2011.
- [14] M.-R. Ra, J. Paek, A. B. Sharma, R. Govindan, M. H. Krieger, and M. J. Neely. Energy-delay tradeoffs in smartphone applications. In *Proc. of ACM MobiSys*, 2010.
- [15] A. Rahmati and L. Zhong. Context-for-wireless: Context-sensitive energy-efficient wireless data transfer. In *Proc. of ACM MobiSys*, 2007.
- [16] C. Raiciu, D. Niculescu, M. Bagnulo, and M. J. Handley. Opportunistic mobility with multipath TCP. In *Proc. of ACM MobiArch*, 2011.
- [17] C. Raiciu, C. Paasch, S. Barre, A. Ford, M. Honda, F. Duchene, O. Bonaventure, and M. Handley. How hard can it be? Designing and implementing a deployable multipath TCP. In *Proc. of USENIX NSDI*, 2012.
- [18] P. J. Rockwell and R. A. Davis. *Introduction to Time Series and Forecasting*. Springer, 1994.
- [19] Samsung Electronics. Samsung Galaxy S3 Android phone specifications. <http://www.samsung.com/us/mobile/cell-phones/SGH-I74>
- [20] A. Schulman, V. Navda, R. Ramjee, N. Spring, P. Deshpande, C. Grunewald, K. Jain, and V. N. Padmanabhan. Bartendr: A practical approach to energy-aware cellular data scheduling. In *Proc. of ACM MobiCom*, 2010.
- [21] wikipedia. Snapdragon (system on chip). [https://en.wikipedia.org/wiki/Snapdragon\\_S4#Snapdragon](https://en.wikipedia.org/wiki/Snapdragon_S4#Snapdragon)